Drools Game of Life Example

David Wallace Croft
david@CroftSoft.com

Presented to the
Dallas Rules Group
2008 Nov 06 Thu

Slides Online

- http://CroftSoft.com/library/tutorials/drools/
-
-
    -

About the Speaker

- David Wallace Croft
  - http://www.CroftSoft.com/people/david/
- Using Drools at work for data cleansing
- Author of “Advanced Java Game Programming”
- Former JUG-Head
  - Silicon Valley Java Users Group
  - Game Developers Java Users Group
- Part-time PhD student UT Dallas, Neuroscience

Download

- http://www.jboss.org/drools/downloads.html
- Drools 4.0.7 Examples
  - Conway's Game of Life
- Drools 4.0.7 Binaries
  - JAR files needed on classpath
  - Includes Drools and third-party dependencies
- Drools 4.0.7 Source Code
  - Step through Drools core during debugging

Maven Build

- Maven build file in root of examples directory
- Maven build appears to be broken
- I attempted to fix by setting repository
    - http://repository.jboss.com/maven2/org/drools/
- But a dependency library missing
- Maven Eclipse plugin
    - http://m2eclipse.codehaus.org/
- New book: "Maven: The Definitive Guide"

Ant Build

- Also has Ant build file
- Copy binaries Drools JAR files into lib directory
  - Copy third-party dependency JAR files too
- No target for Game of Life example
- Insert Ant target code on next slide

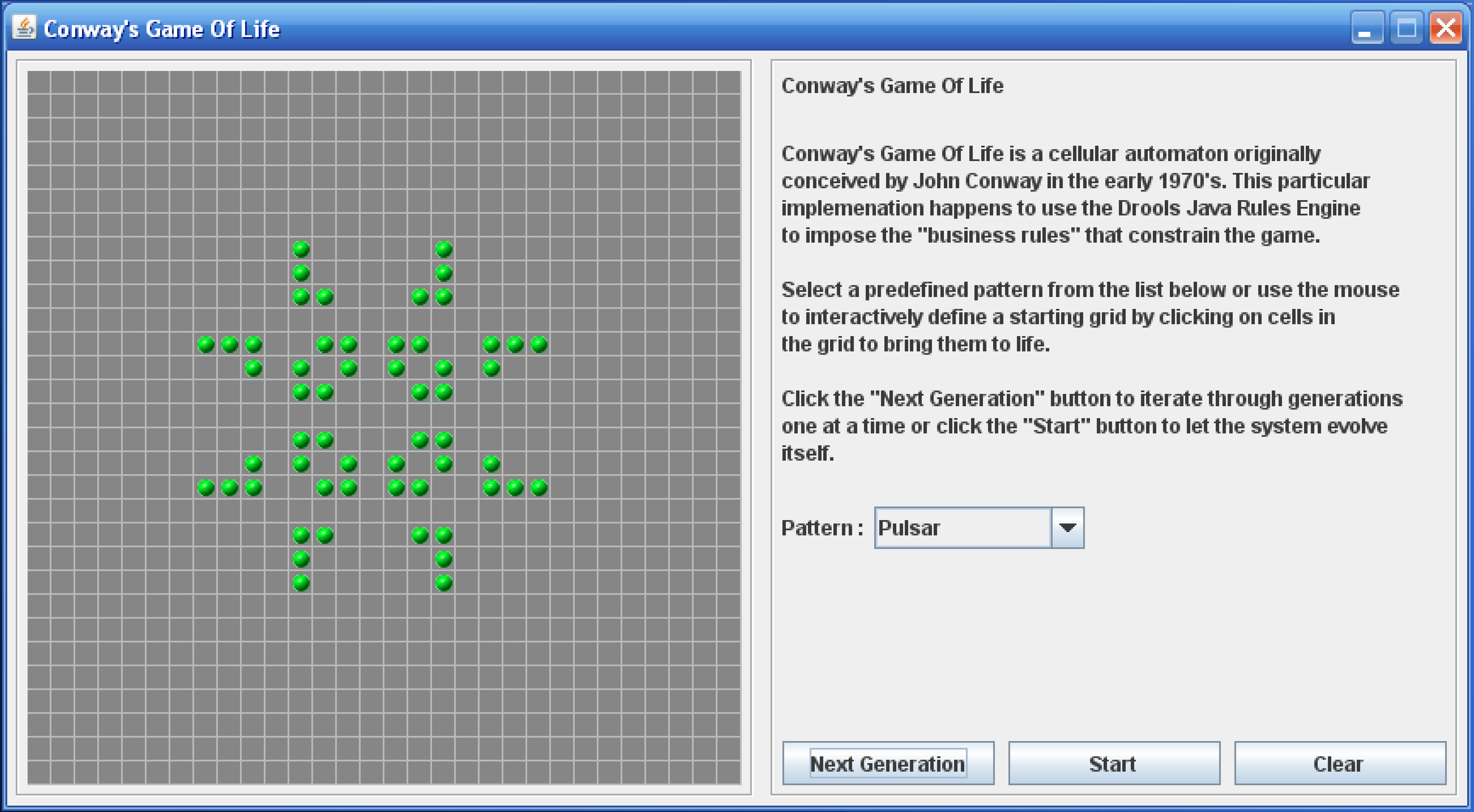
Game of Life Ant Target

```
<target name="run-conway" depends="clean, compile">
  <java fork="true" classname=
    "org.drools.examples.conway.ConwayAgendaGroupRun">
    <classpath>
      <pathelement path="target/classes" />
      <fileset dir="lib" includes="**/*.jar"/>
    </classpath>
  </java>
</target>
```

Conway's Game of Life

- Created by John Conway in 1970
- Cellular automaton
- See Wikipedia entry for history and examples
- Grid of cells, each cell “alive” or “dead”
- When alive and < 2 adjacent live cells, die
- When alive and > 3 adjacent live cells, die
- When dead and = 3 adjacent live cells, live

Demonstration



Conway's Game Of Life

Conway's Game Of Life is a cellular automaton originally conceived by John Conway in the early 1970's. This particular implemenation happens to use the Drools Java Rules Engine to impose the "business rules" that constrain the game.

Select a predefined pattern from the list below or use the mouse to interactively define a starting grid by clicking on cells in the grid to bring them to life.

Click the "Next Generation" button to iterate through generations one at a time or click the "Start" button to let the system evolve itself.

Pattern : Pulsar

Miracle of Life

- Simple rules creates deterministic complexity
- Self-sustaining temporal patterns
- Persistence persists
- Stability is its own reward

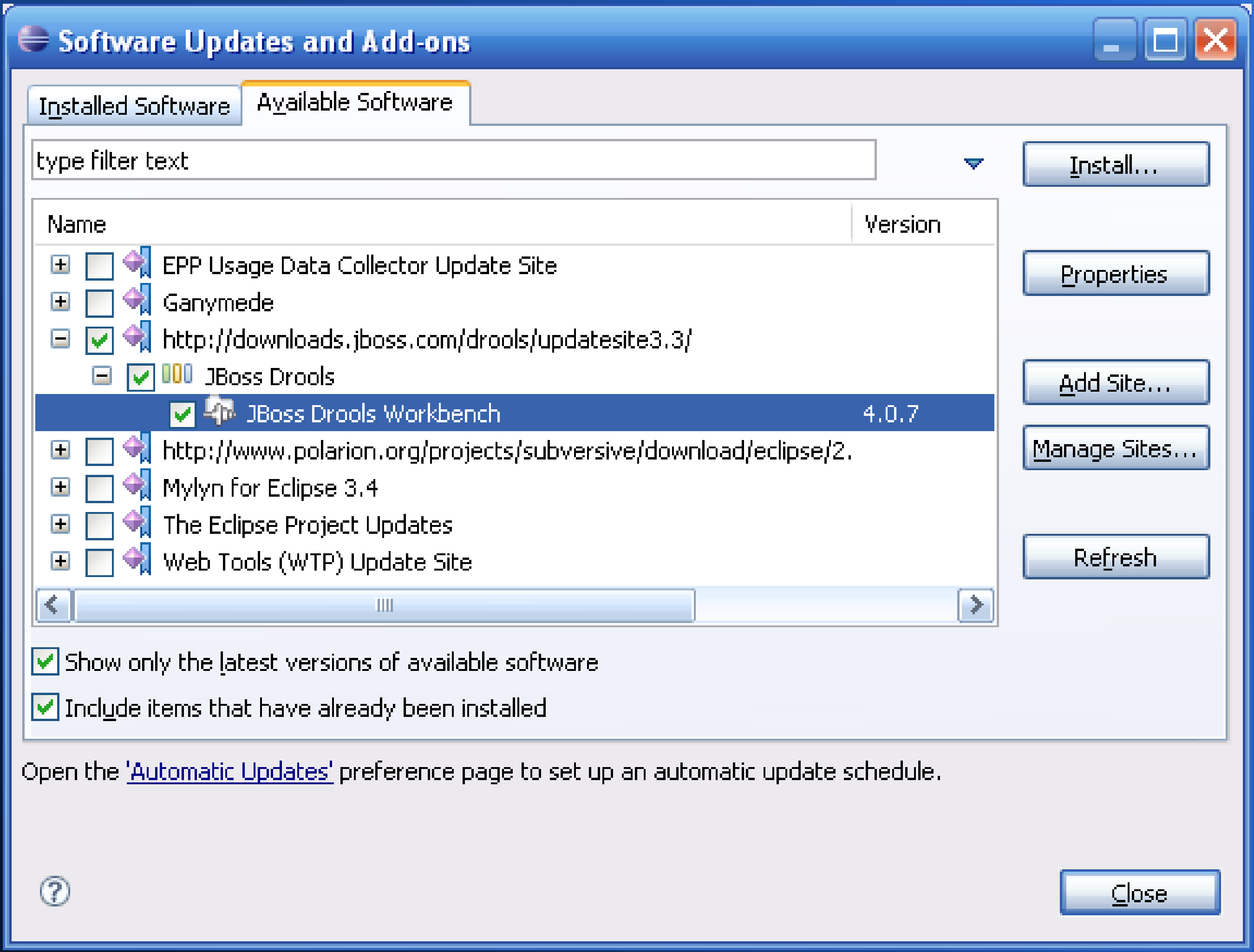
Eclipse

- Drools examples configured for Eclipse Integrated Development Environment (IDE)
  - http://www.eclipse.org/
- Examples for Eclipse 3.3
- But I was able to use Eclipse 3.4

Drools Eclipse Plugin

- Section “3.1.3.3. Installing from the update site”
  - JBoss Rules User Guide - HTML
  - http://www.jboss.org/drools/documentation.html
- Download webpage
  - Section “Drools IDE Update Site”
  - Drools 4.0.7 Eclipse Workbench for Europa 3.3
  - http://www.jboss.org/drools/downloads.html
  - http://downloads.jboss.com/drools/updatesite3.3

Eclipse 3.4 Software Updates



type filter text

Name	Version
⊞ ☐ EPP Usage Data Collector Update Site	
⊞ ☐ Ganymede	
⊟ ✓ http://downloads.jboss.com/drools/updatesite3.3/	
⊟ ✓ JBoss Drools	
✓ JBoss Drools Workbench	4.0.7
⊞ ☐ http://www.polarion.org/projects/subversive/download/eclipse/2.	
⊞ ☐ Mylyn for Eclipse 3.4	
⊞ ☐ The Eclipse Project Updates	
⊞ ☐ Web Tools (WTP) Update Site	

✓ Show only the latest versions of available software

✓ Include items that have already been installed

Open the 'Automatic Updates' preference page to set up an automatic update schedule.

Eclipse Setup

- Add examples directory as a Java project
- Copy binaries Drools JAR files into lib directory
  - Copy third-party dependency JAR files too
- Add JAR files to Eclipse project build classpath
- Delete missing project dependencies
  - Already added to lib directory
- Eclipse thinks there are errors in rules files
  - Fix by deleting extra source directories

Running from Eclipse

- Run from Eclipse for Step Debugging
- Maven directory structure
  - src/main/java/org/drools/examples/conway
- Main classes
  - ConwayAgendaGroupRun.java
  - ConwayRuleFlowGroupRun.java

Eclipse Step Debugging

- F5 - Step Into
- F6 - Step Over
- F7 - Step Return
- F8 - Resume
- Stepping into Drools core will prompt you to locate the Drools core source code

Drools Views

- Set debug breakpoints on fireAllRules() calls
- AgendaGroupDelegate.java
  - init()
  - nextGeneration()
- Window / Show Views / Other / Drools
- If Drools views blank
  - click on session object in debug variables tab
  - data will then load in Drools views

Console Logger

- Append to constructor of AgendaGroupDelegate.java

```
new WorkingMemoryConsoleLogger ( session );
```

- Trace activations
- Slows down processing

Why This Example?

- Does not run until completion when fireAllRules() called
- Applies rules once to current state and stops
- View then updated
- How does it do that?
- If this works, rules can be used in
  - games
  - simulations

Rule-based Virtual Worlds

- Idea from Jack Park, Silicon Valley friend
  - Technical Editor of my book “Advanced Java Game Programming”
  - Editor of book “XML Topic Maps”
- Children learn artificial intelligence (AI) concepts by typing in rules governing entities in virtual world
- I want to use rule-based virtual environments to train computers (neuronal networks, robotics)

Master Method

```
public boolean nextGeneration() {
    session.setFocus( "kill" );
    session.setFocus( "birth" );
    session.setFocus( "reset calculate" );
    session.setFocus( "rest" );
    session.setFocus( "evaluate" );
    session.setFocus( "calculate" );
    session.fireAllRules();
    return session.getAgenda().getAgendaGroup(
      "calculate" ).size() != 0; }
```

Agenda Groups

- Agenda groups
  - Group rules to be processed together
- Setting focus pushes agenda group on stack
- Processed in last in first out (LIFO) order
  - Reverse of order in which setFocus() called
- Used to break rule processing into phases
  - calculate, evaluate, rest, reset, birth, kill

Rule Flow Groups

- Determine order in which rules processed
- Alternative to using rule salience (priority levels) and agenda groups
- Flowchart created using graphical editor
- Drools Game of Life example written to use either rule flow groups or agenda groups
- Only the agenda group configuration studied in this presentation

Activations Forever?

- All activations within an agenda group processed together
- Changes to data lead to new activations
- For self-sustaining temporal patterns, no end
- Activations in agenda groups not on agenda stack are not processed
- setFocus() used to control agenda stack

State and Phase

```
rule "Give Birth"
  agenda-group "evaluate"
  no-loop
when
  theCell: Cell (
    liveNeighbors == 3,
    cellState     == CellState.DEAD,
    phase         == Phase.EVALUATE )
then
     modify ( theCell ) { setPhase ( Phase.BIRTH ) }
```

State Updates

- State not changed immediately
- Marked for change in future processing phase
- Simulations need to be updated in two steps: access and mutate (read and write)
- No state changes until all objects have had a chance to access the state of all other objects that they depend on to determine next state
- Makes it independent of order processed

Phase Tagging

- Rules include condition that checks phase property of object
- Phase property must match agenda group phase currently being processed
- Changing phase property prevents object from being processed again in this phase
- Tags/flags/marks object as being done

Processing Done

```
rule "birth"
  agenda-group "birth"
  no-loop
when
  theCell: Cell ( phase == Phase.BIRTH )
then
  modify ( theCell ) {
    setCellState ( CellState.LIVE ),
    setPhase ( Phase.DONE ) }
end
```

No-Loop

- “When the Rule's consequence modifies a fact it may cause the Rule to activate again, causing recursion. Setting no-loop to true means the attempt to create the Activation for the current set of data will be ignored.” -- Drools doc
- Redundant when used with phase tagging?
- I remarked out all no-loops in the example and it still worked fine

Initialization

```
public void init() {
    this.session.setFocus( "register neighbor" );
    this.session.fireAllRules();
    session.clearAgendaGroup( "calculate" );
}
```

- Establishes neighbor relationships
- Just one phase gets focus
- Resulting activations cleared after

Neighbors

```
rule "register north"
    agenda-group "register neighbor"
when
    $cell: Cell( $row : row > 0, $col : col )
    $north : Cell( row  == ($row - 1), col == $col )
then
    insert( new Neighbor( $cell, $north ) );
    insert( new Neighbor( $north, $cell ) );
end
```

Neighbor Count

```
rule "Calculate Live"
  agenda-group "calculate"
  lock-on-active
when
  theCell: Cell(cellState == CellState.LIVE)
  Neighbor(cell == theCell, $neighbor : neighbor)
then
  modify ( $neighbor { setLiveNeighbors (
    $neighbor.getLiveNeighbors() + 1 ),
    setPhase( Phase.EVALUATE ) }
```

Lock-On-Active

- “when [...] an agenda-group receives the focus any rules that have lock-on-active set to true cannot place activations onto the agenda, the rules are matched and the resulting activations discarded. This is a stronger version of no-loop. It's ideally for calculation rules where you have a number of rules that will modify a fact and you don't want any rule re-matching and firing.” -- Drools doc

Activations Forever

- If you remark out, stuck in activation loop

```
rule "Calculate Live"
  agenda-group "calculate"
  // lock-on-active
when
  theCell: Cell(cellState == CellState.LIVE)
  Neighbor(cell == theCell, $neighbor : neighbor)
then
  modify ( $neighbor { setLiveNeighbors (
```

Conclusion

- Games and simulations need rules that advance the state one step and then stop
- This requires that activations on agenda stop
- Agenda groups can be used to process the rules in phases
- Phase tagging can be used to create conditions that mark data as already being processed in the current phase